Arduino: PID-controlled thermostat

or, how to do things with Arduino without ever becoming an expert

Sean P. Robinson
spatrick@mit.edu

Massachusetts Institute of Technology
Department of Physics

BFY 2012 Workshop — July 26, 2012

Outline

1. Introductions (You, Me, and Arduino)
2. Microcontrollers, microcontroller projects, and how they can help you run your physics teaching lab
3. How to make a project
4. An Exercise, if we have time
(or Homework if we don't)

What we did, and why we did it

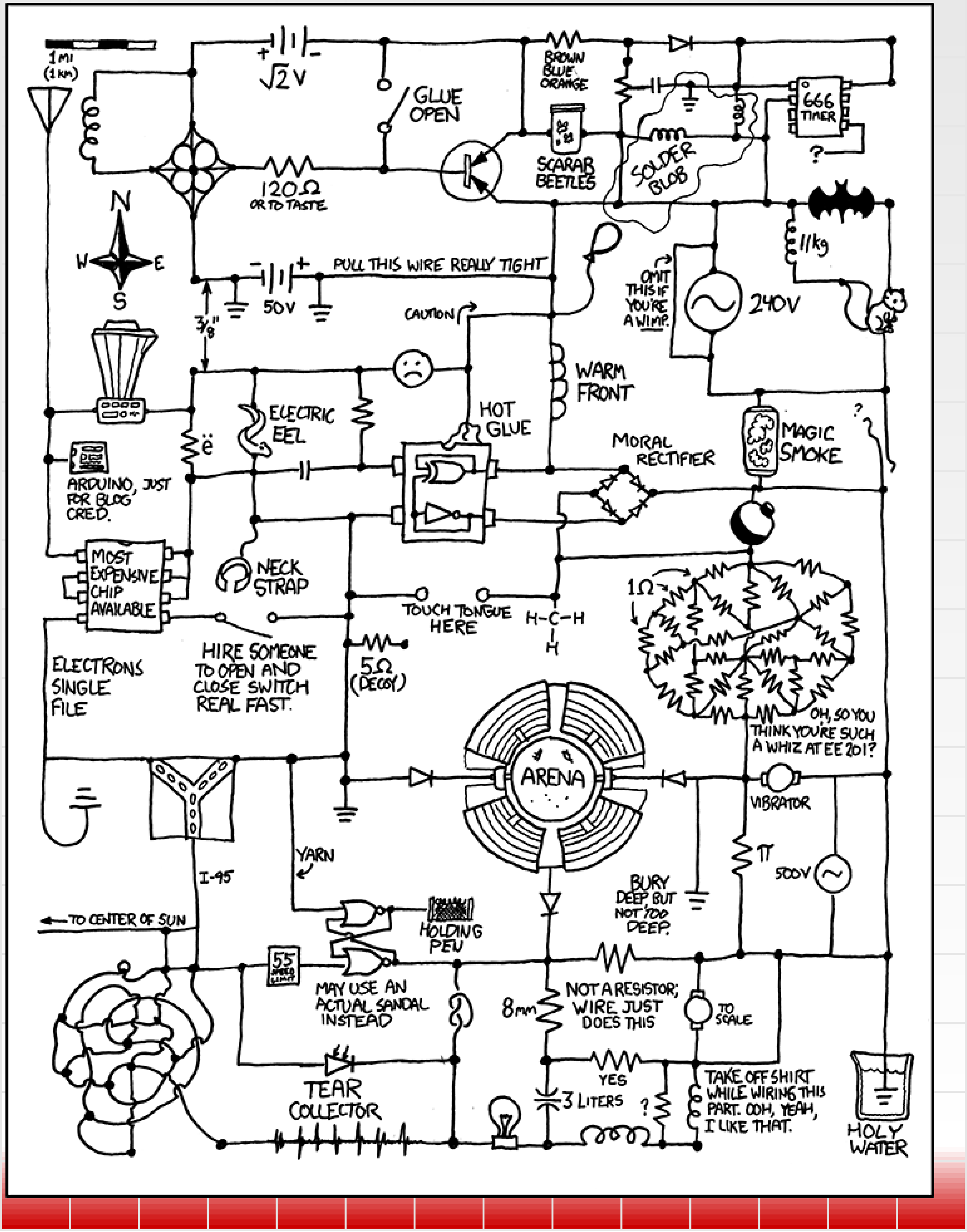
WHAT: We used an Arduino microcontroller to build a fancy thermostat.

- Several experiments use heating ovens with temperature control by setting VARIAC voltage supply to a resistive heating element.
  - Requires trial and error to establish voltage-temperature relation
  - Slow to stabilize (10s of minutes) → trial and error is *tedious*!!
- Use a thermostat (negative feedback towards set point) instead!
- This exercise: let's add challenge by using the PID control algorithm instead of simple (threshold model) thermostat.

WHY: we were looking for a simple project to practice using Arduino in a full build

- Everyone's doing it! (Large user community, including local friends.)
- Inexpensive (open source). Hardware ≈ $25–30, software free.
- They say it's easy. (Much of the user community is nontechnical.)
- Street cred with the cool kids.

XKCD.COM/730/


1 mi
(1 km)
+ √2 V -
GLUE OPEN
BROWN BLUE ORANGE
666 TIMER
?
SCARAB BEETLES
SOLDER BLOB
120Ω OR TO TASTE
N
W
E
S
PULL THIS WIRE REALLY TIGHT
50V
3/8"
CAUTION
OMIT THIS IF YOU'RE A WIMP.
240V
1/kg
WARM FRONT
ELECTRIC EEL
ë
HOT GLUE
MORAL RECTIFIER
MAGIC SMOKE
?
ARDUINO, JUST FOR BLOG CRED.
MOST EXPENSIVE CHIP AVAILABLE
NECK STRAP
TOUCH TONGUE HERE
H-C-H
H
1Ω
ELECTRONS SINGLE FILE
HIRE SOMEONE TO OPEN AND CLOSE SWITCH REAL FAST.
5Ω (DECOY)
OH, SO YOU THINK YOU'RE SUCH A WHIZ AT EE 201?
ARENA
VIBRATOR
π
500V
YARN
I-95
TO CENTER OF SUN
HOLDING PEN
BURY DEEP BUT NOT TOO DEEP.
55 SPEED LIMIT
MAY USE AN ACTUAL SANDAL INSTEAD
8mm
NOT A RESISTOR; WIRE JUST DOES THIS
TO SCALE
TEAR COLLECTOR
YES
3 LITERS
?
TAKE OFF SHIRT WHILE WIRING THIS PART. OOH, YEAH, I LIKE THAT.
HOLY WATER

Microcontrollers: What are they?

Definition

Microcontroller: a small computer, often consisting of a single board running whatever program has been uploaded to it, whose inputs and outputs are analog and/or digital voltage ports. (Usually includes a power input and some kind of serial connection for receiving uploaded programs from some other computer.)

- Can be used like a data acquisition device (*eg* LabJack or NI-DAQ), for low resolution (10-bit ≈ 5 mV resolution) and low bandwidth (250 samples/sec, or up to 10–50 kHz with tricks).
- Can be used as part of a feedback/control system between inputs and outputs.
- Can be used as a programmable voltage source.
- . . . and more!

Examples include BASIC Stamp, Arduino, Cypress (kind of) and others.

Why Arduino?

Arduino is a relatively new — but massively popular — player among microcontrollers.

- Open Source: an appealing moral æsthetic like Linux, Firefox, *etc*
  ⇒ low cost! And, ample resources on the web.
- Popular among artists and the DIY/"maker" crowd
  ⇒ so, you don't need a degree in CS/EE to use Arduino.
  ⇒ and, there's a huge online menu of project ideas and problem solutions.
  ⇒ . . . and that includes a lot of my students.
- Extendable hardware: daughter boards (called "shields") extend the hardware capability, just like software libraries extend the software.

What does a working project with a microcontroller look like?

Well, look at what we have here. . .

This is a feedback-controlled piece of lab equipment. Ingredients:

- an Arduino Uno ($26) for overall control
- a TC4 shield ($30, surface mount components already soldered) for better resolution in reading thermocouple. Developed and sold by gourmet coffee roasting hobbyists, `homeroasters.org` (`http://code.google.com/p/tc4-shield/`)
- solid state relay ($22) to switch power ON and OFF
- LCD display ($6) so we can see what we're doing
- a 12V DC power supply, a type-T thermocouple, bits of wire, a knob, some power cord, and other minor bits from around the lab
- an aluminum box and a plastic cover from around the lab
- the equipment to be controlled

How to start a project

Assuming you have an idea of something to build . . .

Get Arduino software (free) and hardware (cheap)

- Download the Arduino IDE development environment: `arduino.cc/en/Main/Software`
- Buy Arduino hardware. Many suppliers, but we like these guys: `www.adafruit.com/category/17`

Don't get confused by the varieties of Arduino boards. Unless your project is high-performance, you probably want the Arduino Uno.

Check out tutorials!!

- **Getting Started with Arduino** is what it sounds like: `arduino.cc/en/Guide/HomePage`
- Lady Ada's tutorial is the best: `www.ladyada.net/learn/arduino/`
- Take a glance at the Arduino playground: `arduino.cc/playground/projects/ideas`

Someone has probably done this before

. . . or least parts of it. Break problem into constituent parts and then

. . . to the INTERNET!

The joy of open source

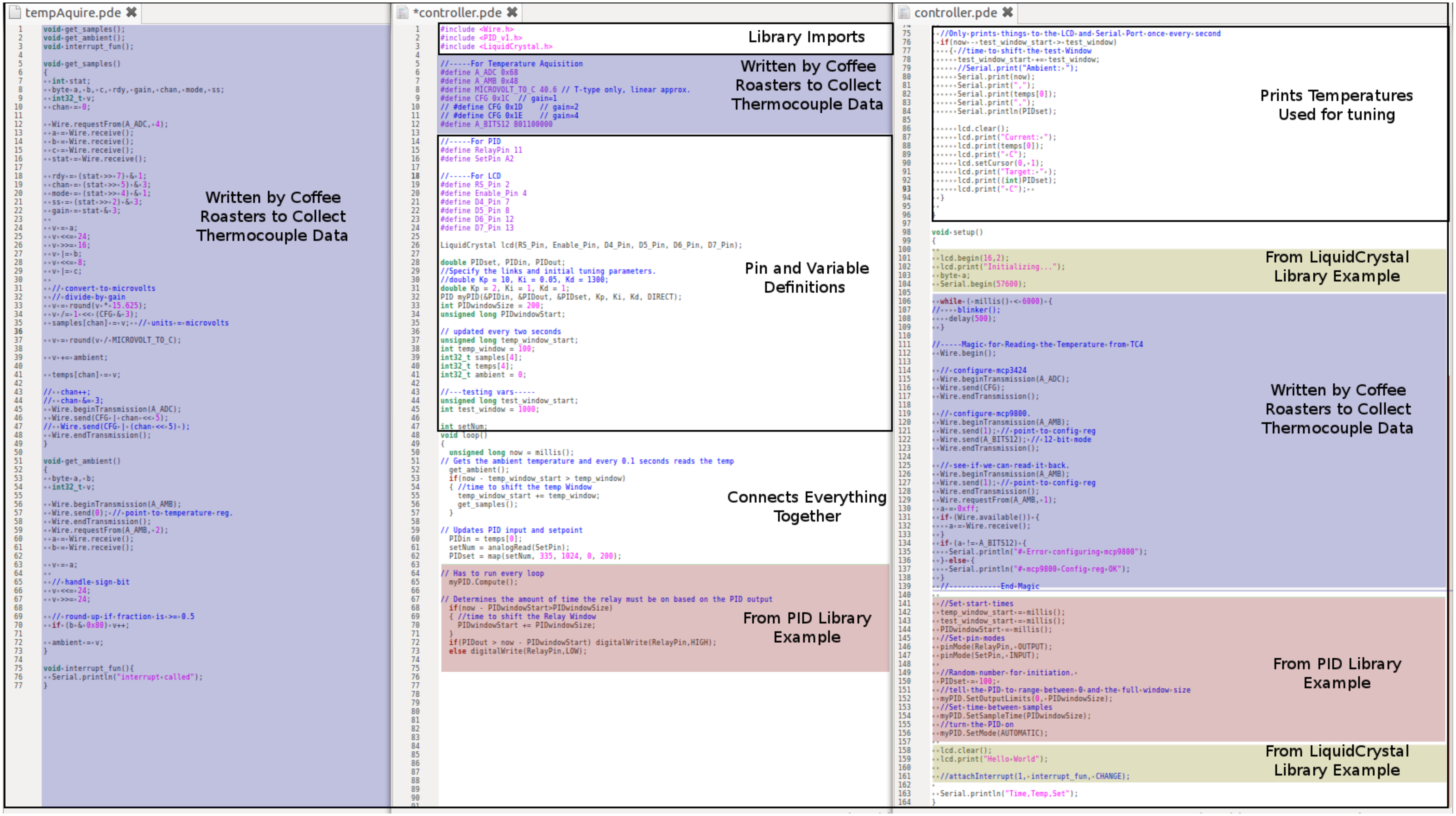
Things to search for

- Whole projects
- Shields that extend hardware capability
- Software libraries
- Code examples
- Circuit diagrams
- Better ideas

The Arduino programming language is basically C++, so programming help is also widely available on the internet and in your neighbor's office.

Use other people's code!

Every Arduino program needs a function named "`void setup()`" and one named "`void loop()`".



tempAquire.pde

Written by Coffee Roasters to Collect Thermocouple Data

```
void get_samples();
void get_ambient();
void interrupt_fun();

void get_samples()
{
  int stat;
  byte a, b, c, rdy, gain, chan, mode, ss;
  int32_t v;
  chan = 0;

  Wire.requestFrom(A_ADC, 4);
  a = Wire.receive();
  b = Wire.receive();
  c = Wire.receive();
  stat = Wire.receive();

  rdy = (stat >> 7) & 1;
  chan = (stat >> 5) & 3;
  mode = (stat >> 4) & 1;
  ss = (stat >> 2) & 3;
  gain = stat & 3;
  
  v = a;
  v <<= 24;
  v >>= 16;
  v |= b;
  v <<= 8;
  v |= c;
  
  // convert to microvolts
  // divide by gain
  v = round(v * 15.625);
  v /= 1 << (CFG & 3);
  samples[chan] = v;  // units = microvolts

  v = round(v / MICROVOLT_TO_C);

  v += ambient;

  temps[chan] = v;

//  chan++;
//  chan &= 3;
  Wire.beginTransmission(A_ADC);
  Wire.send(CFG | chan << 5);
//  Wire.send(CFG | (chan << 5) );
  Wire.endTransmission();
}

void get_ambient()
{
  byte a, b;
  int32_t v;

  Wire.beginTransmission(A_AMB);
  Wire.send(0); // point to temperature reg.
  Wire.endTransmission();
  Wire.requestFrom(A_AMB, 2);
  a = Wire.receive();
  b = Wire.receive();

  v = a;
  
  // handle sign-bit
  v <<= 24;
  v >>= 24;

  // round up if fraction is >= 0.5
  if (b & 0x80) v++;

  ambient = v;
}

void interrupt_fun(){
  Serial.println("interrupt called");
}
```

*controller.pde

Library Imports

```
#include <Wire.h>
#include <PID_v1.h>
#include <LiquidCrystal.h>
```

Written by Coffee Roasters to Collect Thermocouple Data

```
//-----For Temperature Aquisition
#define A_ADC 0x68
#define A_AMB 0x48
#define MICROVOLT_TO_C 40.6 // T-type only, linear approx.
#define CFG 0x1C  // gain=1
// #define CFG 0x1D    // gain=2
// #define CFG 0x1E    // gain=4
#define A_BITS12 B01100000
```

Pin and Variable Definitions

```
//-----For PID
#define RelayPin 11
#define SetPin A2

//-----For LCD
#define RS_Pin 2
#define Enable_Pin 4
#define D4_Pin 7
#define D5_Pin 8
#define D6_Pin 12
#define D7_Pin 13

LiquidCrystal lcd(RS_Pin, Enable_Pin, D4_Pin, D5_Pin, D6_Pin, D7_Pin);

double PIDset, PIDin, PIDout;
//Specify the links and initial tuning parameters.
//double Kp = 10, Ki = 0.05, Kd = 1300;
double Kp = 2, Ki = 1, Kd = 1;
PID myPID(&PIDin, &PIDout, &PIDset, Kp, Ki, Kd, DIRECT);
int PIDwindowSize = 200;
unsigned long PIDwindowStart;

// updated every two seconds
unsigned long temp_window_start;
int temp_window = 100;
int32_t samples[4];
int32_t temps[4];
int32_t ambient = 0;

//---testing vars-----
unsigned long test_window_start;
int test_window = 1000;

int setNum;
```

Connects Everything Together

```
void loop()
{
  unsigned long now = millis();
// Gets the ambient temperature and every 0.1 seconds reads the temp
  get_ambient();
  if(now - temp_window_start > temp_window)
  { //time to shift the temp Window
    temp_window_start += temp_window;
    get_samples();
  }

// Updates PID input and setpoint
  PIDin = temps[0];
  setNum = analogRead(SetPin);
  PIDset = map(setNum, 335, 1024, 0, 200);
```

From PID Library Example

```
// Has to run every loop
  myPID.Compute();

// Determines the amount of time the relay must be on based on the PID output
  if(now - PIDwindowStart>PIDwindowSize)
  { //time to shift the Relay Window
    PIDwindowStart += PIDwindowSize;
  }
  if(PIDout > now - PIDwindowStart) digitalWrite(RelayPin,HIGH);
  else digitalWrite(RelayPin,LOW);
```

controller.pde

Prints Temperatures Used for tuning

```
 //Only prints things to the LCD and Serial Port once every second
 if(now - test_window_start > test_window)
   { //time to shift the test Window
     test_window_start += test_window;
     //Serial.print("Ambient: ");
     Serial.print(now);
     Serial.print(",");
     Serial.print(temps[0]);
     Serial.print(",");
     Serial.println(PIDset);

     lcd.clear();
     lcd.print("Current: ");
     lcd.print(temps[0]);
     lcd.print(" C");
     lcd.setCursor(0, 1);
     lcd.print("Target: " );
     lcd.print((int)PIDset);
     lcd.print(" C");  
 }
  
}

void setup()
{
```

From LiquidCrystal Library Example

```
  
  lcd.begin(16,2);
  lcd.print("Initializing...");
  byte a;
  Serial.begin(57600);
```

Written by Coffee Roasters to Collect Thermocouple Data

```
  while ( millis() < 6000) {
//    blinker();
    delay(500);
  }

//-----Magic for Reading the Temperature from TC4
  Wire.begin();

  // configure mcp3424
  Wire.beginTransmission(A_ADC);
  Wire.send(CFG);
  Wire.endTransmission();

  // configure mcp9800.
  Wire.beginTransmission(A_AMB);
  Wire.send(1); // point to config reg
  Wire.send(A_BITS12); // 12-bit mode
  Wire.endTransmission();

  // see if we can read it back.
  Wire.beginTransmission(A_AMB);
  Wire.send(1); // point to config reg
  Wire.endTransmission();
  Wire.requestFrom(A_AMB, 1);
  a = 0xff;
  if (Wire.available()) {
    a = Wire.receive();
  }
  if (a != A_BITS12) {
    Serial.println("# Error configuring mcp9800");
  } else {
    Serial.println("# mcp9800 Config reg OK");
  }
  //------------End Magic
```

From PID Library Example

```
  //Set start times
  temp_window_start = millis();
  test_window_start = millis();
  PIDwindowStart = millis();
  //Set pin modes
  pinMode(RelayPin, OUTPUT);
  pinMode(SetPin, INPUT);
  
  //Random number for initiation. 
  PIDset = 100; 
  //tell the PID to range between 0 and the full window size
  myPID.SetOutputLimits(0, PIDwindowSize);
  //Set time between samples
  myPID.SetSampleTime(PIDwindowSize);
  //turn the PID on
  myPID.SetMode(AUTOMATIC);
```

From LiquidCrystal Library Example

```
  lcd.clear();
  lcd.print("Hello World");
  
  //attachInterrupt(1, interrupt_fun, CHANGE);
 
  Serial.println("Time,Temp,Set");
}
```

About our temperature controller project: two goals

1. An excuse to learn Arduino
2. Build a cheap, useful temperature controller.

The first goal outweighs the second, so we may make some odd choices.

The PID control loop

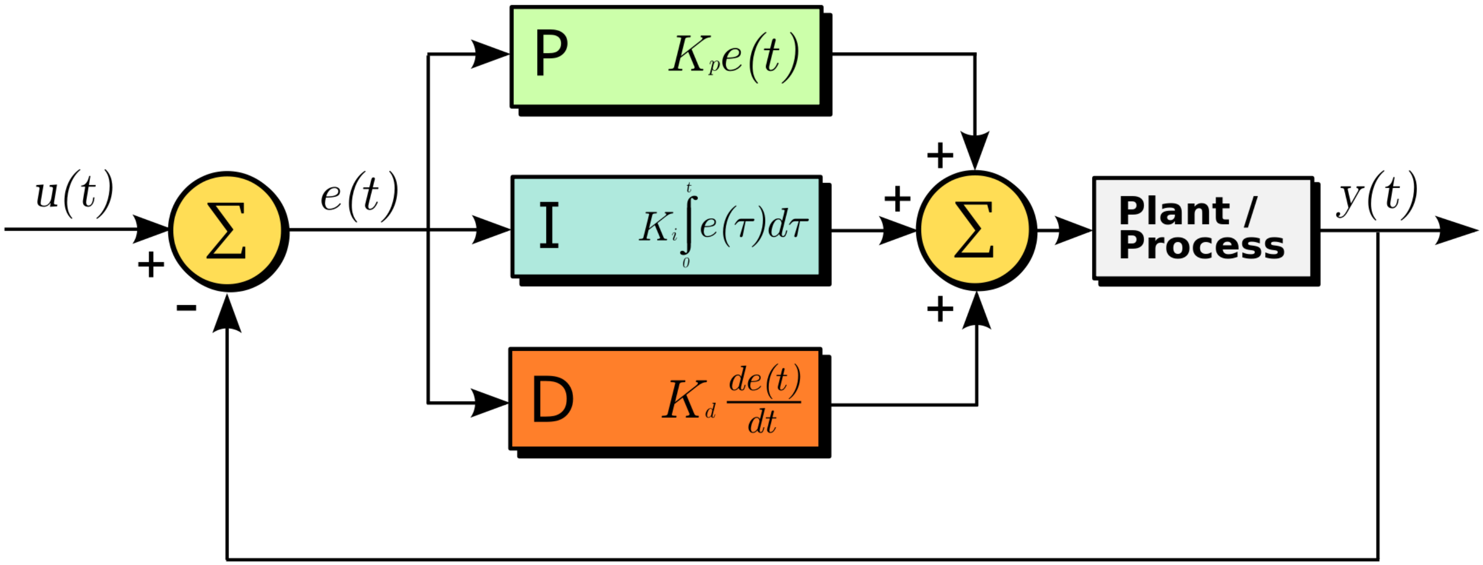
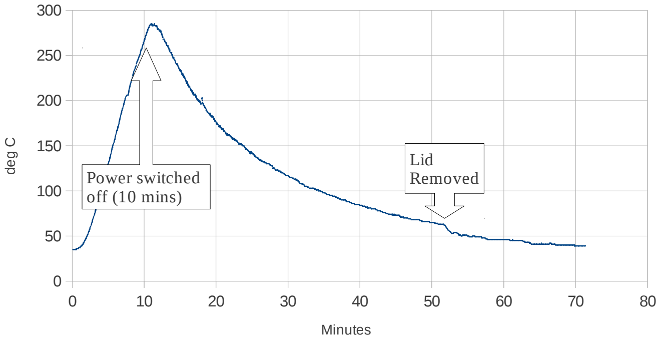
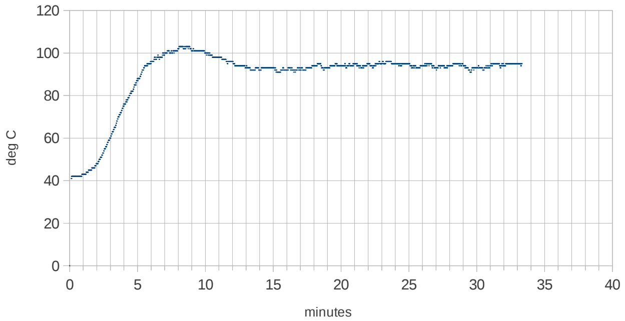


Image source: Wikipedia

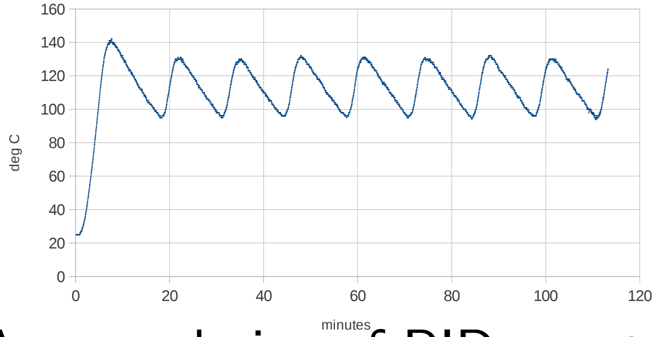
Performance



No regulation ⇒ rapid overheating



$(K_p, K_i, K_d) = (2.5, 1/1.6\ min, 4\ min)$. Very stable, little overshoot, but 5°C of "droop".



A poor choice of PID constants, $(K_p, K_i, K_d) = (2, 1/1\ ms, 1\ ms)$.

- Tuning PID constants is difficult.
- Not every system is a good candidate for PID control.
- Our system happens to be one of them.

Let's think of a project . . .